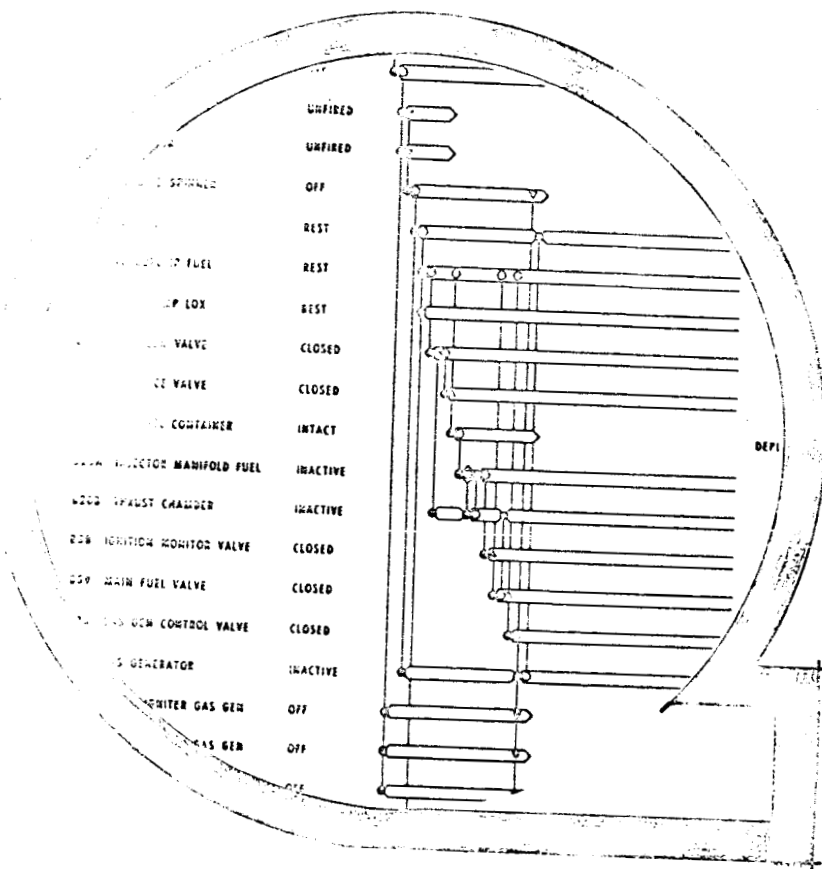


REPRODUCED FROM BEST AVAILABLE COPY



GENERAL SYSTEMS RELATIONSHIP MANAGEMENT SYSTEM

GPO PRICE \$ _____

CFSTI PRICE(S) \$ _____

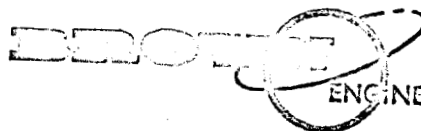
Hard copy (HC) 2.00

Microfiche (MF) 1.50

ff 653 July 65

JUNE 30, 1964

Systems Simulation Section



ENGINEERING COMPANY, INC.
HUNTSVILLE, ALABAMA

N66 26694

(ACCESSION NUMBER)

39
(PAGES)

CR 75034
(NASA CR OR TMX OR AD NUMBER)

(THRU)

(CODE)

08
(CATEGORY)

5QT-2829P

3600 1800 1800

ABSTRACT

26694

This report presents a symbolic language for expressing functional relationships in space vehicle systems. It defines and illustrates the methods of presentation and use of the General Functional Relationship Language System.

TABLE OF CONTENTS

Section		Page
I.	SUMMARY	1
II.	INTRODUCTION	
	A. Authorization for the Project	2
	B. Purpose of the Language System	3
	C. Approach to the Problem	3
	D. General Description of the Report	4
III.	DESCRIPTION	
	A. Major Concepts of the Language System	5
	B. General Description of the Language System	6
	C. Using the Language System	9
	D. Equipment and Materials	20
IV.	COMPUTER PROGRAMS OF THE LANGUAGE SYSTEM .	23
	A. Event Generator Program	25
	B. Output Control Program	26
	C. The Plot Program	29
V.	CONCLUSIONS AND RECOMMENDATIONS	33
VI.	GLOSSARY OF TERMS	34

LIST OF ILLUSTRATIONS

Figure		Page
1	The Language System	5
2	Component Functional Relationships	8
3	The Steps of the General Functional Relationship Language System	9
4	The Data Areas of the Component Information Worksheet	11
5	A Completed Component Information Worksheet	12
6	Data Worksheet Layouts	14
7	The Graphical Sequence of Functional Relationship in an H1 Engine System Start Sequence	18
8	Sequence Diagram Symbols Description	19
9	The Verbal Sequence of Functional Relationships in an H1 Engine System Start Sequence	21
10	Computer Programs of the Language System	23
11	Executive Routine of the Language System Programs . .	24
12	Logic of the Plot Program	32

LIST OF TABLES

Table		Page
1	Data Worksheet Field Code Description	15
2	Recommended Equipment	22
3	Recommended Materials	22

I. SUMMARY.

This report presents the results of a study to provide an automated method of expressing the detailed functional relationships that occur between components as they interact during operation in a space vehicle. A symbolic language was devised (called the General Functional Relationship Language) to describe component data to a computer, enabling it to construct a mathematical/symbolical simulation of the functional relationships within a space vehicle. The output forms of the language are varied easily to yield symbolic representations that lend themselves to analysis of functional relationship problems. Additional research and development are required to integrate this language system with the other language systems in order to construct a computer model of a space vehicle system.

The report includes a discussion of the development of the language system, its symbols, the methods of analyzing component data, the logic of the computer programs, material and equipment requirements, and graphical outputs.

II. INTRODUCTION.

A. Authorization for the Project.

The contract for this system simulation study was awarded to Brown Engineering Co. under NAS8-11027, and the program started approximately 13 months before this report was published. The program objective was to develop two language systems which could describe a portion of a space vehicle of the Saturn class, be used in a digital computer, and be compatible with a third language developed by NASA. (Refer to NASA Internal Note MTP-QUAL-64-1. *) When the three languages are combined, they will provide a "total system definitive statement" which will describe the entire vehicle system. These three language systems, called the General Functional Relationship Language, the Intercomponent Relationship Language, and the Introcomponent Relationship Language, describe respectively, how one component affects another component (their functional relationship), the physical interconnection between components (the intercomponent relationship), and the dynamic behavior of components (the introcomponent relationship).

The General Functional Relationship Language System, the subject of this report, was developed to express the functional interrelationships of the components of the electrical and mechanical or electro-mechanical systems, in a suitable notation, and if possible, to develop techniques for automated troubleshooting of the Saturn vehicle during systems performance tests.

* Modeling of Space Vehicle Component Dynamics Using a Digital Differential Analyzer Method by C.M. Webster (G.E.) and R.W. Foster (NASA).

B. Purpose of the Language System.

The need for the General Functional Relationship Language is apparent when we attempt to analyze and verify the compatibility, adequacy, or redundancy of the detailed functional relationships that occur between components as they interact during operation of a space vehicle. The purpose of the General Functional Relationship Language is to express the functional relationships of a space vehicle, and automatically modify that expression throughout design, manufacture, checkout, and launch phases.

C. Approach to the Problem.

The development of the General Functional Relationship Language started with an examination of documentation pertinent to the functions of the Saturn booster. Mechanical and Electrical schematics, operational sequence charts, functional descriptions, and operations data were analyzed. Particular attention was given to time sequence requirements and cause and effect relationships. The functional components that were established in the design criteria were accepted as the functional entities that would be the basis of the General Functional Relationship Language. The next objective was to determine the possible states-of-being that these components could assume. The final consideration was to determine the functional interrelationships among components. These interrelationships were described by identifying the components that contributed to a change-of-state in a component and to describe their manner of contribution.

In order to avoid the semantic problems of expressing functional relationships to various users, a graphical symbology was developed. A verbal output, comparable in information and meaning, to the graphical output, would be ponderous. Analysis of similar studies enabled us to avoid duplication of study, and provided us with many of the graphic symbols for expressing system functions.

The result was a feasible automated method of simulating the combined functional relationships of all the functional components of the system.

D. General Description of the Report.

The first portion of this report describes the language system. This description includes the major concepts and definitions evolved, the mechanics of transforming the data into the language algorithm, the methods of obtaining outputs, and the necessary language system equipment and materials.

The programs designed to handle the source or engineering data are then discussed. Detailed programming codes are not given in this report, but will be in separate programming manuals. In the conclusion, suggestions are given for potential applications of the functional language system. Special terms used in this report are defined in the glossary.

III. DESCRIPTION.

This description first explains the major concepts of the language system. It then describes the steps required to use the language.

A. Major Concepts of the Language System.

Three concepts must be accepted to understand the language system. First, a special language is used to express functional data. Second, the functional data are constructed into an algorithm. Last, the algorithm is used as a core from which specific outputs are produced. (See figure 1.)

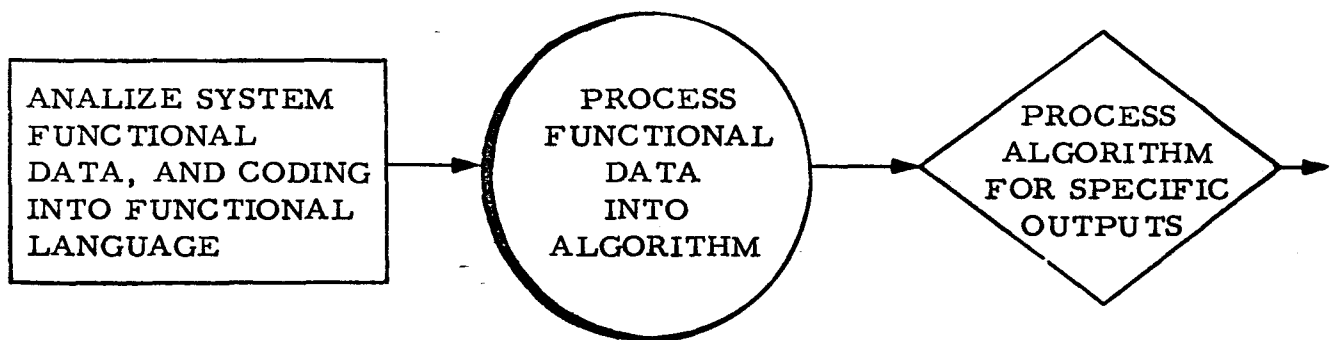


Figure 1. The Language System

1. Language.

The body of the language is composed of words and symbols whose meanings have been standardized in engineering usage. While analyzing the source data and processing it into an algorithm, the actual verbal descriptions of components and functions are used, such as "detonator," "indicator," "firing," and "not on". However, these words are represented by graphical symbols when we produce graphical outputs to illustrate the results of the computer processing.

2. Algorithm.

The algorithm as used in this language system is the words and symbols processed into a fixed arrangement or relationship. As such, it represents the functional interrelationships of all the functional components in the space vehicle system. It is constructed by the computer, and may be modified by the computer to match modifications made to the actual space vehicle system that it represents.

B. General Description of the Language System.

The General Functional Relationship Language description of a space vehicle is concerned only with functions of the vehicle systems. In the design criteria of the space vehicle, its functional requirements are defined. The functional requirements are separated into specific functions which become associated with specific system components. Each component so established is then considered a functional entity of the total system, and referred to as a functional component. It may be a single part or a combination of parts that function independently within a complete operating system but provide a self-contained function necessary for proper system operation.

As a space vehicle system operates, the states-of-being (or conditions), that a component can assume, change. The states-of-being may range from the simple on-off states of a switch to more complex states as empty, filling, and full for a tank. These possible states-of-being are noted and made part of the data of the General Functional Relationship Language.

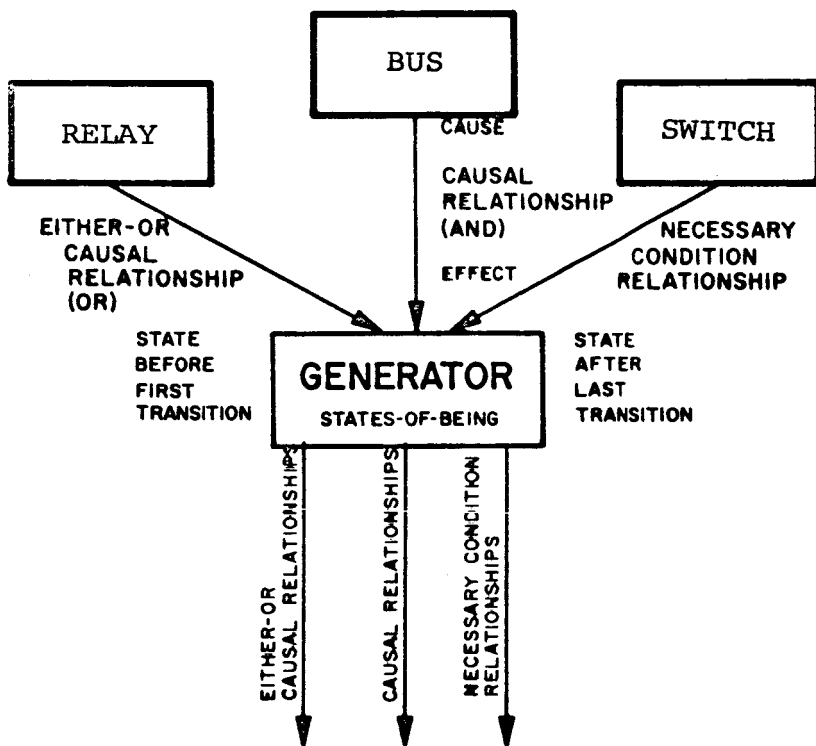
(See figure 2.)

The possible states-of-being of a component may be changed by the internal functioning of the component, or may be changed by the functions of other components. When a change in state in a component is caused by the action of other components, there is a causal relationship between the components. When a change in state in a component is dependent upon some other components being in a certain state, there exists a "necessary condition" relationship. As a component changes states, it causes other components to change state. This, in turn, affects other components. The functional interrelationships are noted for each component and made a part of the data of the General Functional Relationship Language. (See figure 2.)

An "event" occurs when a component changes state. Each time an event occurs in a component there is a specific sequential chain of interrelationships coincident with the event. Said another way, at an exact time a combination of conditions exists in functionally related components to constitute an event. In the graphical output symbology this is illustrated by a vertical line connecting all component conditions of the event.

The data required to describe one component and its functional relationships to a computer are: (1) the component name, (2) possible states-of-being of the component, including initial state, and (3) names of contributing components and their manner of contribution or relationship.

The computer processes these data into the general functional relationships of all components in the space vehicle system.



-COMPONENT
DATA-

- NAME OF COMPONENT
- STATES OF COMPONENT
- CONTRIBUTING COMPONENTS
- CONTRIBUTING RELATIONSHIPS
- RESULTING RELATIONSHIPS

Figure 2. Component Functional Relationships

C. Using the Language System.

The overall mechanics of the General Functional Relationship Language System are explained here in two phases. First, the steps involved in constructing the algorithm are explained in detail. Then the steps required to generate outputs are described. See figure 3 for the complete series of steps.

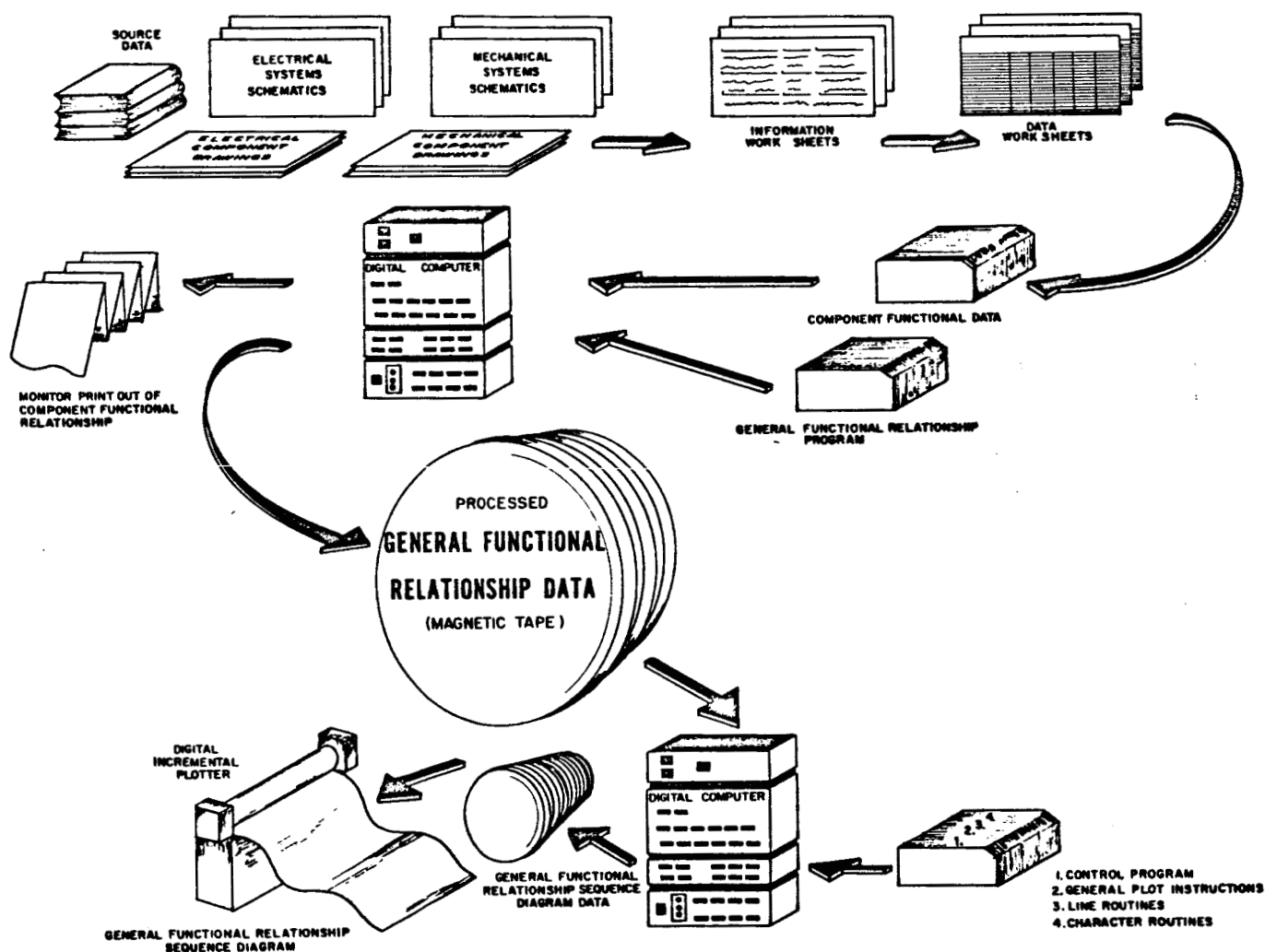


Figure 3. The Steps of the General Functional Relationship Language System

(Step 1). Analysis of the Source Data.

The component functional relationships for a space vehicle system are extracted from the source data of the space vehicle system. The source data may be comprised of electrical or mechanical drawings and schematics, on a system or component level. To determine the minimum functional units of the system, the source data are analyzed, and the components are isolated and named. Each of the states-of-being that can be assumed by the component and all of the conditions necessary for the component to change from one state-of-being to another are determined. The initial and final state of the component are also determined during analysis, and the component data are completed for a component by listing all the other components that have functional relationships with it. The analysis of the source data is performed with the aid of information worksheets. The information worksheets help specify and summarize the components and their relationships prior to the start of data processing.

(Step 2). The Component Information Worksheet.

The Component Information Worksheet enables the engineer to record and work with one component at a time. (See figure 4.) As he reviews the source documentation, he enters the following kinds of data on the worksheets.

- a. Identification of the components.
- b. Possible states-of-being of the component.

- c. Contributing components: identity and manner of contribution. (These are the conditions that need to be satisfied to establish all the functional relationships for the component identified on this worksheet.)

IDENTIFICATION OF COMPONENT											
COMPONENT INFORMATION WORKSHEET		COMPONENT CODE		COMPONENT NAME							
		Finding Number	Drawing Number	Rev. No.	Date	Elec. Sys. Cross Ref.	Mfg'r Code Number	Mfg'r Dwg. No.			
VEHICLE											
STAGE											
SYSTEM											
SUB-SYSTEM											
PREPARED BY:											
Name											
Organization											
DATE											
		State of the Component Prior to the First Transition				State of the Component after Last Transition					
		States of the Component				Time	Start	Time in State	Time at End		
		I				ate			of State		
		II									
		III									
		IV									
		V									
		POSSIBLE STATES-OF-BEING									
		REMARKS:									
		CONTRIBUTING COMPONENTS AND MANNER OF CONTRIBUTION									
CONTRIBUTING COMPONENTS		Component Code	State	Component Code	State	Component Code	State	Component Code	State	Component Code	State
I	Causes (AND's)										
	(OR's)										
	Necessary Condition										
II	Causes (AND's)										
	(OR's)										
	Necessary Condition										
III	Causes (AND's)										
	(OR's)										
	Necessary Condition										
IV	Causes (AND's)										
	(OR's)										
	Necessary Condition										
V	Causes (AND's)										
	(OR's)										
	Necessary Condition										

Figure 4. The Data Areas of the Component Information Worksheet

The data entered on the Component Information Worksheet are in verbal form and essentially in the same form as found in the source data. Figure 5 illustrates a worksheet that contains all the data necessary to describe one component.

COMPONENT INFORMATION WORKSHEET		COMPONENT CODE		COMPONENT NAME				Sheet <u>1</u> of <u>1</u>			
VEHICLE SA-5		B22-1		GAS GENERATOR COMBUSTOR ASSY.							
STAGE 2-1		Finding Number	Drawing Number	Rev. No.	Date	Elec. Sys. Cross Ref.	Mfg'r Code Number	Mfg'r Dwg. No.	REMARKS:		
SYSTEM ENGINE		B22-1	307360	C	8/1/88	NONE					
SUB-SYSTEM ENGINE #1		State of the Component Prior to the First Transition				State of the Component after Last Transition					
PREPARED BY: Name GEORGE SMITH		INACTIVE				BURNING					
Organisation R-GUAL											
DATE 4-2-84											
		States of the Component				Time at Start of State	Time in State	Time at End of State			
		I PRE-HEATING				3	16	19			
		II BURNING (BOOTS TRAP)				19	8	27			
		III									
		IV									
		V									
CONTRIBUTING COMPONENTS		Component Code	State	Component Code	State	Component Code	State	Component Code	State	Component Code	State
I Causes (AND's)		B20-1	BURNING								
(OR's)											
Necessary Condition											
II Causes (AND's)		B20-1	BURNING			B42C-1	HEATING	B42A-1	HEATING	B42B-1	HEATING
(OR's)		B23-1	OPEN	B49-1	OPEN						
Necessary Condition											
III Causes (AND's)											
(OR's)											
Necessary Condition											
IV Causes (AND's)											
(OR's)											
Necessary Condition											
V Causes (AND's)											
(OR's)											
Necessary Condition											

Figure 5. A Completed Component Information Worksheet

(Step 3). Data Worksheets and Punched Cards.

The data on the Information Worksheet are transferred to Data Worksheets. The format of the data worksheets and the format of the punched cards are identical. Key punch operators code the data from the data worksheets directly onto punch cards. The punch cards then contain the component functional relationship data in a form that is fed directly into the digital computer.

There are five data worksheet layouts; and five corresponding punch card designs. The first and second sheets contain the identification of the component, one copy of both are required to identify one component. Sheet three contains the data describing all the possible states-of-being of a component, and only one sheet is needed for a component. Sheets four and five contain the data for all contributing components and their manner of contribution. Many of these sheets are required for components with many functional relationships. See figure 6 for the layout of the field codes of the five data worksheets. Refer to table 1 for an explanation of the field codes of the five data worksheets.

Table 1. Data Worksheet Field Code Descriptions

FIELD CODE COLUMNS	ITEM	DESCRIPTION
WORKSHEET ONE		
1-6	Component Code	A maximum of six characters is used as a unique identification of the component.
7-66	Component Name	A maximum of sixty characters is used for the name of the component.
WORKSHEET TWO		
7-12	Find Number	Find number of the component.
13-20	Drawing Number	Drawing number of the component.
21-26	Rev. Number	Revision number of the drawing.
27-32	Date	Date of drawing. Use month, day, and year, i.e., 051164 for May 11, 1964.
33-38	Elec. Sys. Cross Ref.	Electric System Cross reference number of the component.
39-44	Manufacturer Code Number	Manufacturer's code number for the component.
45-50	Manufacturer Dwg. Number	Manufacturer's drawing number for the component.
WORKSHEET THREE		
7-8	No. of States	The total number of states that the component will have in the duration of the running of the program. If the number is less than ten, a lead zero must be used.
9-10, 11-12 13-14, 15-16 17-18	States of the Components	A maximum of two digits is used as a code to denote the different states that the component will have. These states must be given in the order in which they will occur.

Table 1. Data Worksheet Field Code Descriptions (Continued)

FIELD CODE COLUMNS	ITEM	DESCRIPTION
WORKSHEET FOUR		
7-12	Duration of Transition	The length of time in which the transition takes place.
13-14	No. of Contr. Components	The number of components that contribute to the transition is written. If the number is less than ten, a lead zero must be used.
15-20, 24-29, 33-38, 42-47, 51-56, 60-65, 69-74	Contributing Component	The code name of the contributing component is written.
21, 30, 39, 48, 57, 66, 75	Relation	The nature of the relation of the component to the transition is written.
22-23, 31-32, 40-41, 49-50, 58-59, 67-68, 76-77	Required State	A maximum of two digits is used to denote the state in which the component must be before the transition can take place.
WORKSHEET FIVE		
1-2	State Code	Number assigned to a particular state. If the number is less than ten, a lead zero must be used.
3-26	State Name	Name of a state-of-being that a component may have.

(Step 4). Constructing the Algorithm.

The component functional data cards are read into the computer along with a control program (The Event Generator Program). The individual functional components are then processed by the computer into the General Functional Relationship Algorithm. For an explanation of the control programs, refer to the next section of this report.

(Step 5). The Completed Algorithm.

The product of this first computer run is an algorithm. It is a statement (in digital computer logic) of the combined operation of all of the functional components of the space vehicle system in terms of functional components and sequenced events. The computer provides this statement on magnetic tape. It may concurrently provide a line printer output that shows what data are being processed in the computer at any particular time during the computer run.

(Step 6). Outputs of the Algorithm.

One of the possible outputs that can be obtained from computer processing of the General Functional Relationship data is the sequence diagram. The sequence diagram is a graphic representation of the operation of functional components of a space vehicle system. (See figure 7.) Functional components and their changing states are illustrated for a simulated operation or flight of the vehicle system. The complete set of functional relationships of the individual components is plotted against a relative time scale. The completed sequence diagram, for the Saturn systems study is contained

on a drawing 7 feet by 27 feet. The diagram contains all the functional relationships from initial power turn-on to separation in flight of the upper stages of the Saturn space vehicle.

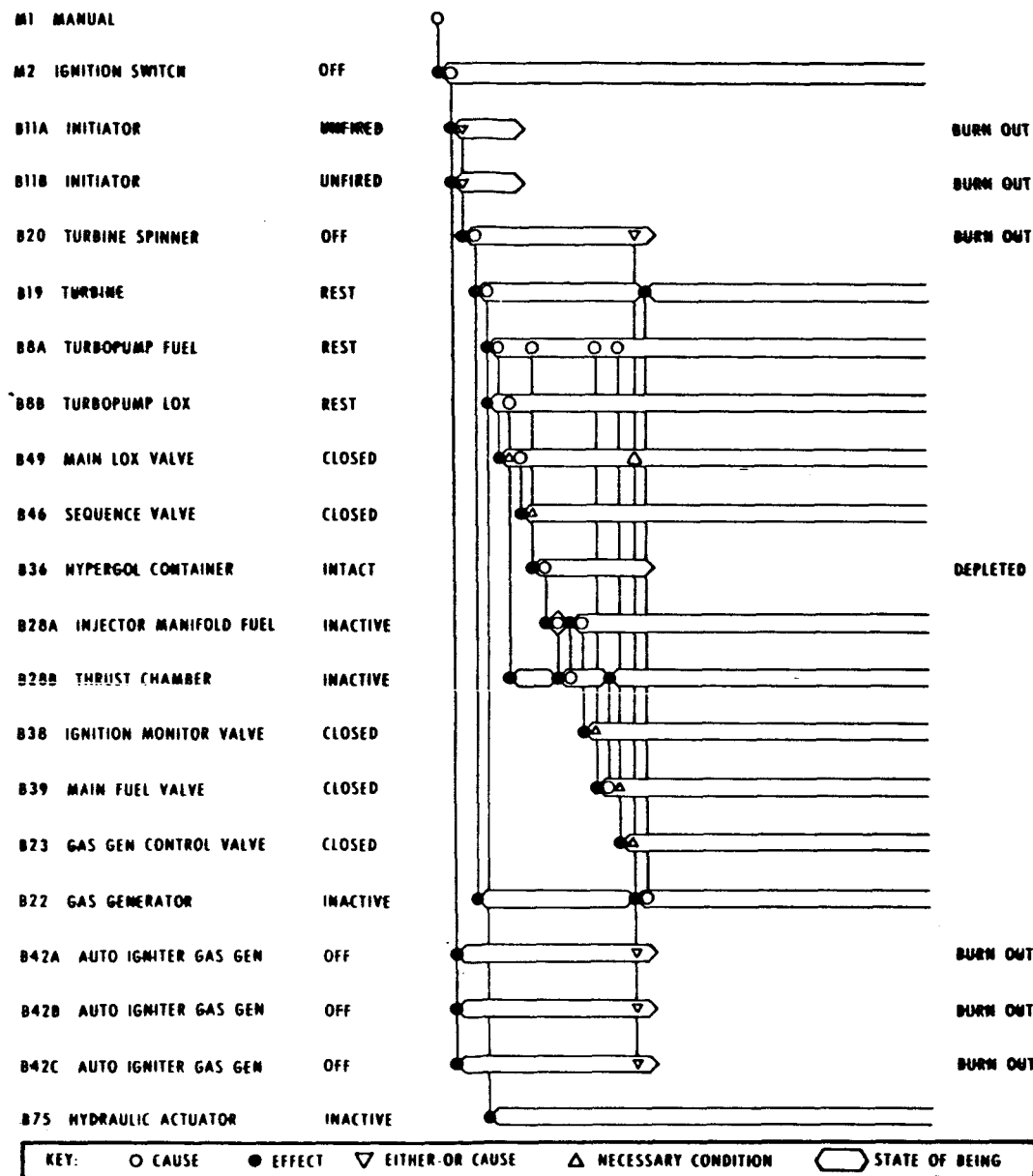


Figure 7. The Graphical Sequence of Functional Relationships in an H1 Engine System Start Sequence

To produce the sequence diagram, the general functional relationship algorithm is fed back into the digital computer with the Output Control Program. The Output Control Program consists of plotting instructions and sequence diagram symbols. (See figure 8.) From this computer run, a Plot Routine is formed on magnetic tape. These instructions are the point-to-point directions for the Digital Incremental Plotter. (A digital incremental plotter is an automated machine that can reproduce graphically the results of the computer analysis, which is on the magnetic tapes.)







WORKSHEET DESCRIPTION	DIAGRAM SYMBOL
CAUSE: A causing operation.	
EFFECT: An effected operation beginning a transition.	
An effected operation terminating a transition.	
EITHER-OR CAUSE: An either-or causing operation.	
NECESSARY CONDITION: A necessary or provided that condition.	
STATE-OF-BEING: The relative time that an event remains in a state. (Possible states are: ON, OPEN, BURNING, ACTIVE, etc.)	

Figure 8. Sequence Diagram Symbols Description

The magnetic tapes of the Plot Routine are mounted on a tape reader. As the tape reader reads the Plot Routine, the plotter draws the sequence diagram. The functional components are listed one at a time, from top to bottom. As they change from their initial states, the duration of the states is drawn from left to right. The functional relationship events, which connect the components to operating time, are drawn in by the plotter as it moves from top to bottom and left to right, through the entire operation of the space vehicle.

Another product of the algorithm is a verbal printout of the sequence of functional relationships. Figure 9 shows, in verbal form, technical data identical to that given in the sequence diagram of the H1 Engine System Start Sequence (Figure 7).

D. Equipment and Materials.

The main equipment required for the General Functional Relationship Language system is a large scale digital computer with both data processing and scientific computation capabilities. Materials required for the language system mainly are magnetic tapes and display material. Tables 2 and 3 list recommended equipment and materials.

EVENT NO. 1
THE MANUAL M1 CAUSES THE IGNITION SWITCH M2 TO BE ON .

EVENT NO. 2
THE IGNITION SWITCH M2 CAUSES THE INITIATOR B11A TO BE FIRING .
THE INITIATOR B11B TO BE FIRING THE AUTO IGNITER GAS GEN B42A AND
TO BE HEATING THE AUTO IGNITER GAS GEN B42C TO BE HEATING .

EVENT NO. 3
EITHER THE INITIATOR B11A OR THE INITIATOR B11B CAUSE
THE TURBINE SPINNER B20 TO BE BURNING .

EVENT NO. 4
THE TURBINE SPINNER B20 CAUSES THE TURBINE B19 TO BE SPINNING AND
THE GAS GENERATOR B22 TO BE PREHEATING .

EVENT NO. 5
THE TURBINE B19 CAUSES THE TURBOPUMP FUEL B8A TO BE SUPPLYING PRESSURE .
THE TURBOPUMP LOX B8B TO BE SUPPLYING PRESSURE AND
THE HYDRAULIC ACTUATOR B75 TO BE PUMPING .

EVENT NO. 6
THE TURBOPUMP FUEL B8A CAUSES THE MAIN LOX VALVE B49 TO BE OPEN .

EVENT NO. 7
THE TURBOPUMP LOX B8B CAUSES THE THRUST CHAMBER B28B TO BE FLOWING LOX
PROVIDED THE MAIN LOX VALVE B49 IS OPEN .

EVENT NO. 8
THE MAIN LOX VALVE B49 CAUSES THE SEQUENCE VALVE B46 TO BE OPEN .

EVENT NO. 9
THE TURBOPUMP FUEL B8A CAUSES THE HYPERGOL CONTAINER B36 TO BE SUPPLYING HYPERGOL
PROVIDED THE SEQUENCE VALVE B46 IS OPEN .

EVENT NO. 10
THE HYPERGOL CONTAINER B36 CAUSES THE INJECTOR MANIFOLD FUEL B28A
TO BE PRESSURIZED .

EVENT NO. 11
THE INJECTOR MANIFOLD FUEL B28A CAUSES THE THRUST CHAMBER B28B
TO BE PRE IGNITING .

EVENT NO. 12
THE THRUST CHAMBER B28B CAUSES THE INJECTOR MANIFOLD FUEL B28A
TO BE INCREAS. PRESSURE .

EVENT NO. 13
THE INJECTOR MANIFOLD FUEL B28A CAUSES THE IGNITION MONITOR VALVE B38
TO BE OPEN .

EVENT NO. 14
THE TURBOPUMP FUEL B8A CAUSES THE MAIN FUEL VALVE B39 TO BE OPEN
PROVIDED THE IGNITION MONITOR VALVE B38 IS OPEN .

EVENT NO. 15
THE MAIN FUEL VALVE B39 CAUSES THE THRUST CHAMBER B28B TO BE BUILDING UP THRUST .

EVENT NO. 16
THE TURBOPUMP FUEL B8A CAUSES THE GAS GEN CONTROL VALVE B23
TO BE SUPPLYING FUEL PROVIDED THE MAIN FUEL VALVE B39 IS OPEN .

EVENT NO. 17
EITHER THE TURBINE SPINNER B20 OR THE AUTO IGNITER GAS GEN B42A .
THE AUTO IGNITER GAS GEN B42B TO BE BURNING THE AUTO IGNITER GAS GEN B42C CAUSE
IS SUPPLYING FUEL AND THE MAIN LOX VALVE B49 IS OPEN .
PROVIDED THE GAS GEN CONTROL VALVE B23

EVENT NO. 18
THE GAS GENERATOR B22 CAUSES THE TURBINE B19 TO BE AUTO SPINNING .

Figure 9. The verbal sequence of Functional Relationships in an H1

Engine System Start Sequence

Table 2. Recommended Equipment

Equipment	Purpose
1. CARD PUNCH	Punch data and program cards.
2. DIGITAL COMPUTER a. Card reader b. Tape unit	Read data and program into computer. Record/reproduce data, program algorithm, display instructions. Print out monitor information.
3. DIGITAL INCREMENTAL PLOTTER.	Plot sequence diagram and typographic outputs.
4. VISUAL DISPLAY EQUIPMENT (A CRT or as determined by need and state-of-the-art)	Visually display schematic and typographic outputs.

Table 3. Recommended Materials

Materials	Purpose
1. SOURCE DATA	Provide raw information to language system.
2. INFORMATION WORKSHEET	Allow source data to be extracted and analyzed.
3. DATA WORKSHEETS	Allow data from Information Worksheet to be put into format for card punch.
Materials	Purpose
4. PUNCH CARDS	Contain <ul style="list-style-type: none"> a. Data. b. Programs. c. Routines.
5. MAGNETIC TAPES	Contain <ul style="list-style-type: none"> a. Data. b. Programs c. Algorithms. d. Display instructions.
6. LINE PRINTER PRINTOUTS	Contain monitor printouts.
7. DIGITAL INCREMENTAL PLOTTER PLOTS	Provide sequence diagram and typographical plots of outputs.

IV. COMPUTER PROGRAMS OF THE LANGUAGE SYSTEM.

Three computer programs are required to process raw engineering data into a finished graphical representation of the general functional relationships of a space vehicle system.* The first is the Event Generator Program. The second is the Output Control Program. The third is the Plot Program. (See figure 10.)

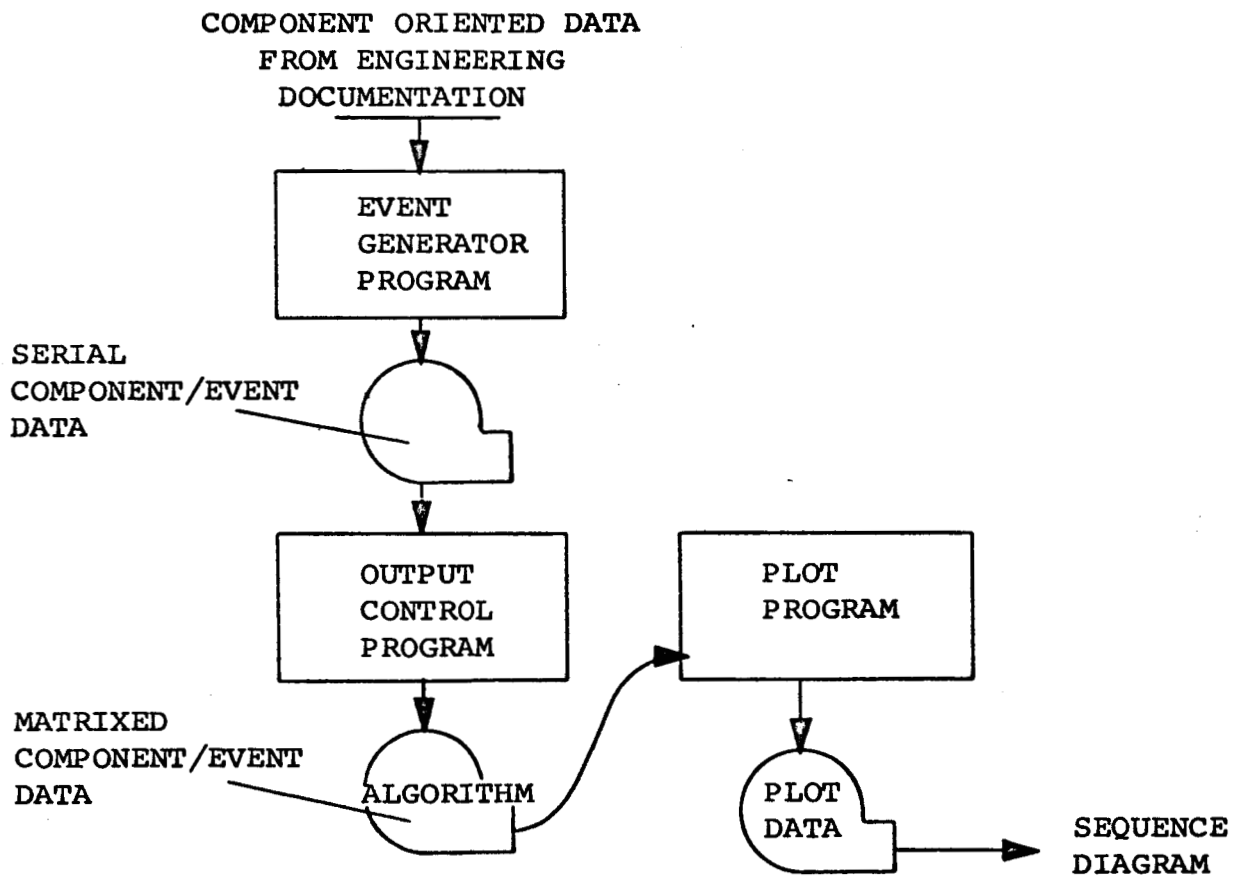


Figure 10. Computer Programs of the Language System

- * In some system applications of the General Functional Relationship Language, the input data have already been partially processed into a form in which the events are known. When this event oriented data is the input to the language system, a Sequence Diagram Generator Program is all that is used to construct the algorithm. Refer to the Sequence Diagram Generator Program Handbook for an explanation of that program.

The Event Generator and Output Control programs are too large to fit into a computer at one time. Therefore the programs have been divided into four parts or links, which are controlled by an executive program. (See figure 11.) The first three links combine the functional relationships into events and then determine the order in which the events take place. These first three links are the Event Generator Program. Link four arranges the data into a form from which a sequence diagram may be obtained. Link four is the Output Control Program.

All of the component data are input into link one. Link one combines Functional Relationships in a manner that defines the events.

Link two determines the numeric order in which the events take place.

Link three rearranges the event data in the order in which the events take place.

Link four, utilizing the component and event data, generates the grid for the sequence diagram.

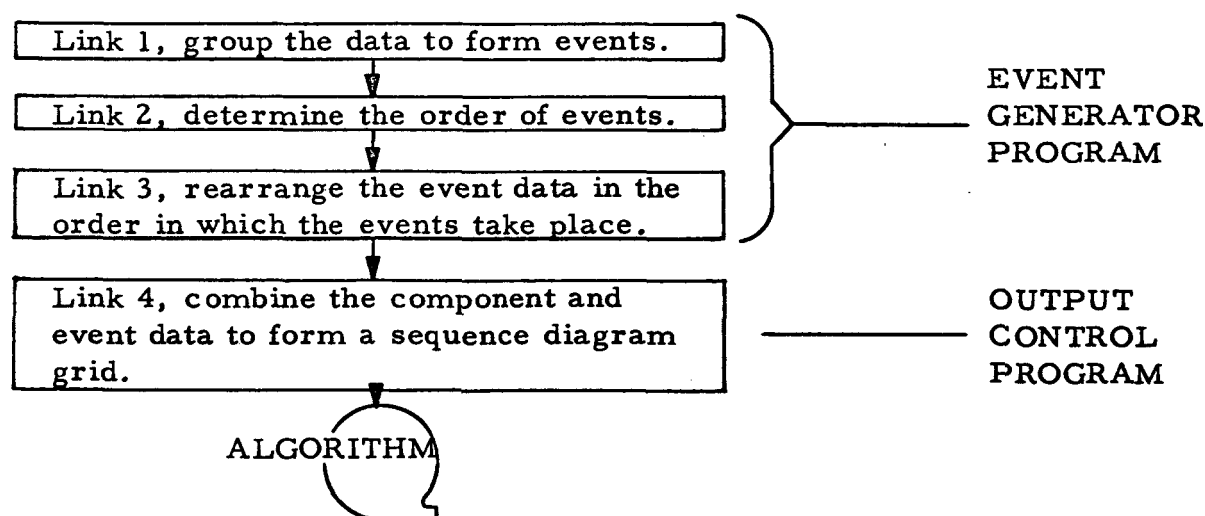


Figure 11. Executive Routine of the Language System Programs.

A. Event Generator Program.

The Event Generator Program determines the order of events from basic data taken from engineering documents.

The basic data are furnished for each component independently of all other components. These data consist of the name of the component, the component code, the states-of-being that the component will have during the time of the checkout, and a description of how other components cause this component to change from one state of being to another. This description is called a general functional relationship and contains the code names of the components that contribute to the change of state, the required states-of-being of these components, and their relationship to the change of state.

A general functional relationship is input for a component. This component is defined as an "effected" component of the relationship. A general functional relationship and all of its "effected" components define an event.

The initial states for all components are input into the PRESENT STATE TABLE of the program. A counter in the program is set equal to zero. The required states-of-being of the contributing components of each event are compared with their corresponding states in the PRESENT STATE TABLE. If all the states are the same, the event takes place. The counter is incremented by one, and the event is assigned the number in the counter (Link one).

When the event takes place, the present states of the effected components are replaced by the next states-of-being of these components. This

process is repeated until all of the events have been assigned a number (Link two).

The event data are rearranged by the computer so that the assigned event numbers are in ascending order (Link three).

The output of this program is a data tape containing the component and event data in serial form.

Refer to the General Functional Relationship Language Event Generator Program Handbook for a detailed explanation of this program.

B. Output Control Program.

The Output Control Program combines the event and component data so that a grid is developed for a sequence diagram output. The component functional relationship input data are encoded and stored in the computer. The program processes the data by extracting one set of data at a time, and comparing it to the remaining stored data. As the functional component relationship data are processed and satisfied, the computer constructs the algorithm grid (Link four). When the relationship data for every functional component have been processed, the algorithm is complete. The contributing and resulting relationships of the components then form a grid which can be traced in either direction by the computer.

The X-axis of the grid represents the event data with one square assigned to each event. The Y-axis of the grid represents the component data with four squares allocated to each component. The first square contains the vertical event lines. The second and fourth squares contain the

horizontal transition lines and the vertical event lines when the component is inactive. The third square contains the identification of the component, the symbols which represent the relationship between the component and its associated events, and the vertical event lines when the component is inactive.

The output of this program is a data tape containing the component and event data in a matrix form to which must be added plot routines to draw a sequence diagram. Refer to the General Functional Relationship Language Event Generator Program Handbook for a detailed explanation of this program.

An example of the construction of a grid from typical event and component data is as follows:

A sequence diagram may be loosely defined as consisting of (1) a set of symbols which show how the components and events are related, (2) vertical lines connecting these symbols, and (3) two horizontal lines drawn from the event which activates a state of a component to the event that deactivates the state of the component.

The following matrix operation is an analagous representation of the preparation of a sequence diagram of NC components and NE events from known component and event data.

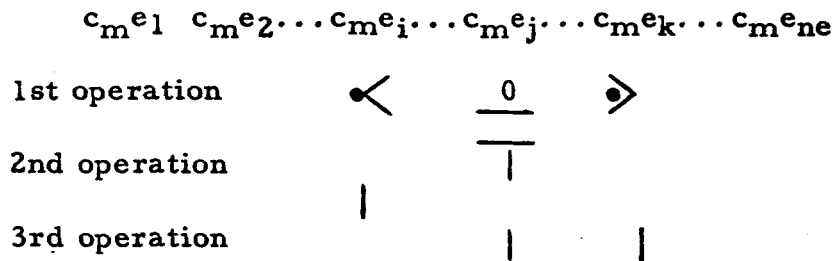
$$\begin{bmatrix} e_1 e \dots e_i \dots e_j \dots e_k \dots e_{ne} \\ \\ \\ c_L \\ \\ \\ c_m \\ \\ \\ c_n \\ \\ \\ c_{nc} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \dots \\ c_L \\ \dots \\ c_m \\ \dots \\ c_n \\ \dots \\ c_{nc} \end{bmatrix} \begin{bmatrix} c_1^{e_1} c_1^{e_2} \dots c_1^{e_i} \dots c_1^{e_j} \dots c_1^{e_k} \dots c_1^{e_{ne}} \\ c_2^{e_1} c_2^{e_2} \dots c_2^{e_i} \dots c_2^{e_j} \dots c_2^{e_k} \dots c_2^{e_{ne}} \\ \dots \\ c_L^{e_1} c_L^{e_2} \dots c_L^{e_i} \dots c_L^{e_j} \dots c_L^{e_k} \dots c_L^{e_{ne}} \\ \dots \\ c_m^{e_1} c_m^{e_2} \dots c_m^{e_i} \dots c_m^{e_j} \dots c_m^{e_k} \dots c_m^{e_{ne}} \\ \dots \\ c_n^{e_1} c_n^{e_2} \dots c_n^{e_i} \dots c_n^{e_j} \dots c_n^{e_k} \dots c_n^{e_{ne}} \\ \dots \\ c_{nc}^{e_1} c_{nc}^{e_2} \dots c_{nc}^{e_i} \dots c_{nc}^{e_j} \dots c_{nc}^{e_k} \dots c_{nc}^{e_{ne}} \end{bmatrix}$$

The following three operations are performed for the m th row.

- 28

3. Straight line symbols are assigned to all of the elements which lie on the ith, jth, and kth columns between elements $c_e e_i$ and $c_m e_i$, $c_e e_j$ and $c_n e_j$, and $c_m e_k$ and $c_n e_k$ respectively.

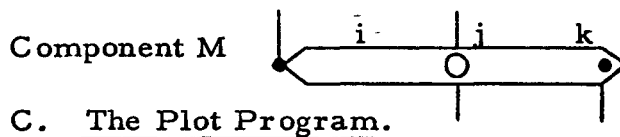
The mth row would look like:



The following is a verbal description of the sequence diagram interpretation of matrix results.

Event e_i causes component c_m to become activated. Event e_j is related in some manner to components c_e , c_m , and c_n . Event e_k causes component c_m to become deactivated.

In sequence diagram the resultant matrix would look like:



The program designed to produce the tapes which instruct the plotter to draw graphical outputs is called the Plot Program. This program adds the plot instructions to the General Functional Relationship Language algorithm and examines the algorithm to determine where to add plot instructions. It reads the symbol code from the algorithm data tape and then automatically generates the plot instructions to produce plot instruction tape. The algorithm data tape is a series of code letters representing the symbols and locations of symbols in a sequence diagram. The Plot instruction data tape

contains the codes necessary to instruct the plotter to draw the graphical output.

There are three kinds of plot instructions:

1. Coordinate locations (which tell the computer the positions for the various lines and characters) for the digital incremental plotter.
2. Line routines, which tell the digital incremental plotter how to draw a line.
3. Character routines, which tell the plotter how to draw the alphabetic, numeric, and special-character symbols.

The logic of the Plot Program is as follows (refer to logic flow diagram, figure 12):

The computer reads the General Functional Relationship Language algorithm data tape until it recognizes a symbol (READ SYMBOL CODE FROM DATA TAPE).

It then determines whether the data tape is ended (END OF FILE) . If it is not the end of the data tape (NO), it goes on to look up the set of coordinates for that symbol in the symbol table (LOOK UP COORDINATES FOR SYMBOL IN SYMBOL TABLE).

When the coordinate locations are established, it look up the line, drawing routine to generate lines to connect the coordinates (LOOK UP LINE DRAWING ROUTINE TO GENERATE LINES TO CONNECT COORDINATES) and generates plot codes to draw the lines or characters (GENERATE PLOT CODE TO DRAW LINE). The computer then determines whether the coordinate was the last coordinate from the symbol table for

that line or character (LAST COORDINATE OF SYMBOL). If it was not the last coordinate (NO), the program goes back to the symbol table and gets another coordinate. If it was the last coordinate (YES), the program then goes on the write the tape (WRITE TAPE).

When the tape is written, the program goes back to read the next symbol code from the data tape (READ SYMBOL CODE FROM DATA TAPE).

This process continues until all symbols are read from the algorithm data tape (END OF FILE). When the end of the data tape is reached (YES), the program exits to END.

The plot instruction data tape produced by this program is fed into the digital incremental plotter to produce the graphical output.

Refer to the General Functional Relationship Language System Graphical Output Program Handbook for a detailed explanation of this program.

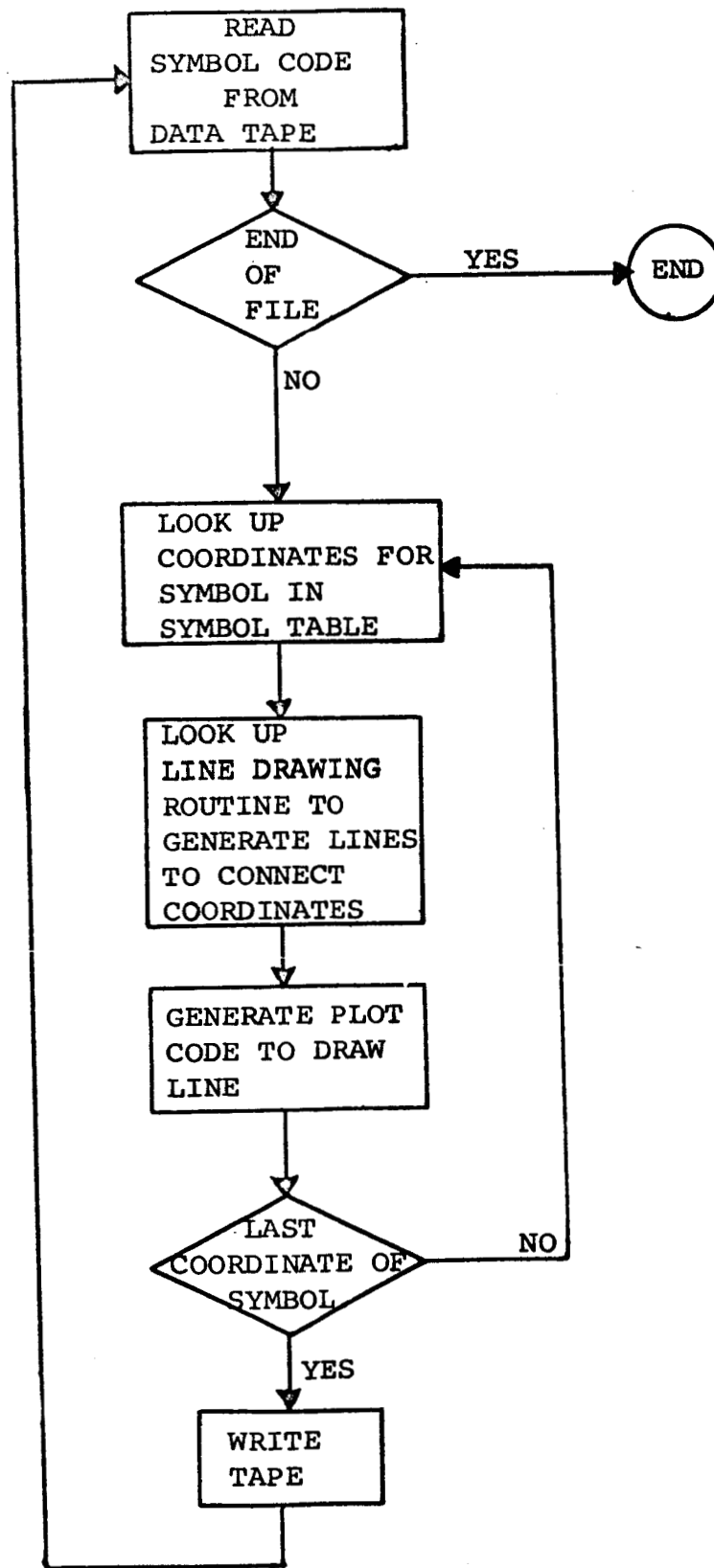


Figure 12. Logic of the Plot Program

V. CONCLUSIONS AND RECOMMENDATIONS.

With the language system developed, space vehicle system functions can be processed by a digital computer. At this time, specific recommendations for the potential uses of the language system can not be given.

Some areas of possible application of the system are:

- (1) Troubleshooting and Fault Isolation
- (2) Automatic Test and Checkout
- (3) Simulated Flight.
- (4) Design Analysis
- (5) Equipment Development
- (6) Operating Rehearsals
- (7) Procedures Generation

Additional study is recommended to determine the areas in which the language system can be used. In order to handle the functional data for a space vehicle system of the size and complexity of the Saturn class, a large-scale digital computer with both data processing and scientific capabilities is required.

VI. GLOSSARY OF TERMS.

1. Algorithm - A set of symbols in a fixed arrangement or relationship.
2. Cause - An operation of a component that causes a resulting change in other components.
3. Character Routines - The computer sub-programs for generating alphabetic and numeric characters and symbols for the digital incremental plotter.
4. Component - A part, or combination of parts, that is functionally independent within a complete operating system, but provides a self-contained function necessary for proper system operation.
5. Component Oriented Data - Data that are acquired and arranged by relating the data to a specific component.
6. Coordinate Locations - The computer sub-programs for generating coordinates for the digital incremental plotter.
7. Digital Incremental Plotter - A machine for plotting drawings from digital computer output data tapes. The heart of the machine is a pen that draws in increments along x and y coordinates.
8. Effect - An operation of a component that results from a change in another component.
9. Event - The specific sequential chain of interrelationships coincident with a component change of state.
10. Event Oriented Data - Data that are acquired and arranged by relating the data to a specific event.

11. Executive Routine - A computer master program for controlling the operation of a group of computer programs.
12. Functional Relationship - The relationships that occur between components as they functionally interact during operation.
13. Language - The group of words and symbols used to describe the function relationships of a system.
14. Language System - The set of rules, procedures, and methods devised to construct and use the algorithm.
15. Line Routines - The computer sub-programs for generating lines for the digital incremental plotter.
16. Links - The phases of operation of a computer master program, each phase completing a specific program operation, arranged in a series such as the links in a chain.
17. Operation - The effect of an event on a component.
18. Sequence Diagram - A diagram that graphically depicts the sequential relationships that occur between components as they functionally interact during operation.
19. States-of-Being - The operational conditions that a component can assume.
20. Symbol Routines - The computer sub-programs for generating graphical symbols for the digital incremental plotter.
21. Total System Definitive Statement - The three computer language systems, i.e., the General Functional Relationship Language, the Intercomponent Relationship Language, and the Introcomponent Relationship Language.